
COSE474-2022F: Final Project

Dynamic Timestep Sampling for Diffusion Models

2018320207 공인호

1. Introduction

디퓨전 모델 $\mathbf{s}_\theta(\mathbf{x}_t, t)$ 는 생성 모델 중 하나로, forward-SDE로 인해 결정되는 perturbed data distribution $p_t(\mathbf{x}_t)$ 의 Score-function $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ 을 예측하고 이를 통해 가우시안 노이즈인 \mathbf{x}_T 로부터 $\mathbf{x}_{T-1}, \dots, \mathbf{x}_0$ 을 샘플링하는 모델이다 (Song et al., 2021).

기존 디퓨전 모델들은 고정된 timestep $t \in \{T, \dots, 0\}$ 를 거쳐 데이터를 샘플링한다. 하지만 \mathbf{x}_t 가 $p_t(\mathbf{x}_t)$ 에 항상 알맞게 형성되는지에 대한 의문이 생겼고, $T = 1$ 이라는 극단적인 케이스를 대입해 생각해 보았을 때 T 가 작은 경우 그렇지 못할 것이라 추측하였다.

Chen et al. (Chen et al., 2022) 또한 T 가 작은 경우 hyper parameter ξ 에 대해 모델의 input 으로 \mathbf{x}_t 와 t 가 아닌 $t + \xi$ 를 사용하는 asymmetric timestep interval 을 제안하였고, 모델의 성능을 향상시켰다. 이는 T 가 작은 경우 x_t 가 $p_t(\mathbf{x}_t)$ 보다 $p_t(\mathbf{x}_{t+\xi})$ 에 더 알맞았다고 해석할 수 있다.

이에 본 프로젝트는 T 가 작은 상황에서, 더 나은 퀄리티의 이미지를 생성하는 샘플링 방법을 찾는 것을 목표로 한다.

Perturbed data \mathbf{x}_t 로부터 t 를 예측하는 모델 Time-Predictor 와 이를 사용한 TP Sampling 을 제안하고, CIFAR-10 Dataset 에서 모델 성능이 향상됨을 확인하였다.

2. Method

2.1. Preliminary

VP-SDE 는 아래와 같이 정의된다 (Song et al., 2021).

$$d\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x} + \sqrt{\beta(t)}d\mathbf{w} \quad (1)$$

이에 대응하는 perturbed kernel $p(\mathbf{x}_t | \mathbf{x}_0)$ 는 아래와 같다.

$$p(\mathbf{x}_t | \mathbf{x}_0) = N(\mathbf{x}_t; \mathbf{x}_0 e^{-\frac{1}{2} \int_0^t \beta(s) ds}, [1 - e^{-\int_0^t \beta(s) ds}] \mathbf{I}) \quad (2)$$

Algorithm 1 TP Training

```

Input: Dataset  $X$ 
repeat
     $\mathbf{x}_0 \sim X$ 
     $t \sim uniform(0, 1)$ 
     $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
     $\mathbf{x}_t = \mathbf{x}_0 e^{-\frac{1}{2} \int_0^t \beta(s) ds} + [1 - e^{-\int_0^t \beta(s) ds}] \epsilon$ 
    Take a gradient step on  $\mathcal{L}_\gamma = \|t - g_\gamma(\mathbf{x}_t)\|^2$ 
until converge is true

```

Continous Time Diffusion Model $\mathbf{s}_\theta(\mathbf{x}_t, t)$ 은 $t \in [0, 1]$ 에 대해 perturbed data distribution $p_t(\mathbf{x}_t)$ 의 score function 을 예측한다 (Song et al., 2021).

$$\mathbf{s}_\theta(\mathbf{x}_t, \sigma_t) \approx \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) \quad (3)$$

위 model 은 아래의 Denoising Score Matching 을 통해 학습 가능하다 (Vincent, 2011).

$$\theta^* = \arg \min_\theta E_{t \sim U(0,1)} \lambda(t) E_{p(\mathbf{x}_0)} E_{p(\mathbf{x}_t | \mathbf{x}_0)} \{ \| \mathbf{s}_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t | \mathbf{x}_0) \|^2 \} \quad (4)$$

2.2. Time-Predictor

본 프로젝트는 perturbed data \mathbf{x}_t 를 입력 받아 t 를 예측하는 모델 Time-Predictor $g_\gamma(\mathbf{x}_t)$ 을 학습시킬 것을 제안한다. 이는 MSE Loss 를 통해 학습 가능하다.

$$\gamma^* = \arg \min_\gamma E_{t \sim U(0,1)} \{ \|t - g_\gamma(\mathbf{x}_t)\|^2 \} \quad (5)$$

학습 알고리즘은 Algorithm 1 에서 확인 가능하다.

2.3. Time Predict Sampling

또한 본 프로젝트는 Time-Predictor 를 이용한 새로운 샘플링 방법 Time Predict Sampling (TP Sampling) 을 제안한다. 이는 기존 샘플링 방법 (EulerMaruyamaPredictor (Song et al., 2021))에서 T 와 interval 은 고정시킨채, t 를 $g_\gamma(\mathbf{x}_t)$ 로 대신한 것이다. Figure 1 은 TP Sampling 을 모식화 한 것이다. 각 샘플링 방법에 대한 알고리즘은 Algorithm 2, 3 에서 확인 가능하다.

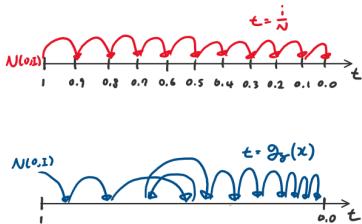


Figure 1.

Algorithm 2 Original Sampling

```

Input: Total timestep  $T$ 
 $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$ 
for  $i = T$  to 1 do
     $t = \frac{i}{T}$ 
     $dt = -\frac{1}{T}$ 
     $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ 
     $(drift) = -\frac{1}{2}\beta(t)\mathbf{x} - \beta(t)\mathbf{s}_\theta(\mathbf{x}, t)$ 
     $(diffusion) = \sqrt{\beta(t)}$ 
     $\mathbf{x} = \mathbf{x} + (drift) dt + (diffusion) \sqrt{-dt} \epsilon$ 
end for

```

3. Experiments

Time-Predictor $g_\gamma(\mathbf{x}_t)$ 는 디퓨전 모델과 독립적으로 학습 가능하기 때문에, pretrained 된 디퓨전 모델을 사용할 수 있다. 본 프로젝트는 DDPM++(cont.) (Song et al., 2021) 을 baseline 으로 잡고, $T = 10$ 일 때 Time-Predictor 및 TP sampling 에 대한 실험을 진행하였다.

3.1. Dataset

본 프로젝트는 CIFAR-10 (Krizhevsky et al., 2009) 데이터셋을 사용하였다. 32x32 사이즈의 10개의 클래스를 가진 이미지 60000 장으로 구성되어 있으며, 학습 데이터는 50000 장, 테스트 데이터는 10000 장이다.

Algorithm 3 TP Sampling

```

Input: Total timestep  $T$ 
 $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$ 
for  $i = T$  to 1 do
     $t = g_\gamma(\mathbf{x})$ 
     $dt = -\frac{1}{T}$ 
     $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ 
     $(drift) = -\frac{1}{2}\beta(t)\mathbf{x} - \beta(t)\mathbf{s}_\theta(\mathbf{x}, t)$ 
     $(diffusion) = \sqrt{\beta(t)}$ 
     $\mathbf{x} = \mathbf{x} + (drift) dt + (diffusion) \sqrt{-dt} \epsilon$ 
end for

```

Table 1. Sampling Method에 따른 FID Score

SAMPLING METHOD	FID SCORE(\downarrow)
ORIGINAL ($T = 1000$)	2.40
ORIGINAL ($T = 10$)	343.35
TP SAMPLING ($T = 10$)	144.88

3.2. Metric

생성된 이미지의 Qualitative comparison 을 위해 Frechet Inception Distance (FID) metric (Heusel et al., 2017)을 사용하였다. 이는 이미지 분포 사이 거리를 측정하는 지표로, 샘플링 데이터의 분포가 원본 데이터 분포와 얼마나 유사한지 알 수 있다. FID score 가 낮을수록 생성된 이미지가 실제 데이터에 가깝다는 것을 의미한다.

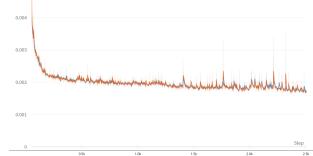


Figure 2. Time-Predictor train(blue)/valid(red) loss

3.3. Results

Figure 2 은 Time-Predictor 의 train/valid loss 를 보여준다. 최종 Valid loss 는 0.0016 으로, MSE loss function 을 고려하였을 때 모델의 output 이 평균적으로 ± 0.04 의 오차를 가짐을 알 수 있다. 비록 loss 가 충분히 작진 않지만, $T = 10$ 일 때 한 스텝의 time interval 이 0.1 이기 때문에 TP Sampling 을 사용할 수 있다.

TP Sampling 에 대한 ablation 결과는 table 1 에서 확인할 수 있다. $T = 10$ 인 경우, TP Sampling 을 적용함으로써 FID score 가 343.35 에서 144.88 로 향상되었다. 또한, Song et al. (Song et al., 2021) 이 발표한 baseline 의 성능이 2.41 이었던 것을 통해, 샘플링 코드에 문제가 없음을 확인 가능하다.

각 Sampling Method 에 따라 생성된 이미지는 Figure 3, 4, 5 에서 확인 가능하다. 이는 생성한 이미지 중에서 랜덤으로 16장을 선택한 것이다.

Figure 3 과 Figure 4 를 비교해보았을 때 기존 Sampling 방법에서는 $T = 10$ 만으론 노이즈가 아직 충분히 제거되지 않고, 심지어 이미지의 class 조차 확인이 불가능함을 알 수 있다. 하지만 이에 반해 Figure 5 는 Figure 4 보다 비교적 노이즈가 적으며, 몇몇 이미지에 대해서는 Figure 3 과 견줄 정도로 좋은 샘플링 퀄리티를 보여주고 있다.

Figure 6 는 TP Sampling ($T = 10$) 중 발생하는 이

미지 데이터 x_t 와 이에 대한 Time-Predictor 의 출력 $g_\gamma(x_t)$ 을 보여준다.

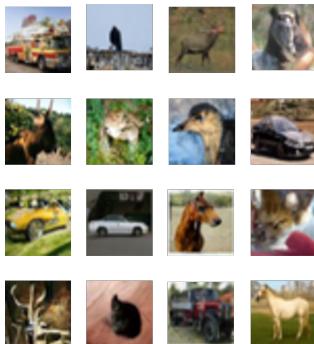


Figure 3. Original Sampling 을 통해 생성낸 이미지 ($T = 1000$)



Figure 4. Original Sampling 을 통해 생성낸 이미지 ($T = 10$)

3.4. Implementation Details

Time-Predictor 의 backbone 모델로는 foundation model 인 Efficient-Net v2-L 을 사용하였다. 이 때 기존 1000 개의 output 을 가지던 마지막 층의 FC layer 를 1개의 output 을 출력하도록 모델 구조를 변경하였다. FC layer 를 제외하고는 ImageNet 으로 학습된 Pretrained weights 를 사용하였으며 모든 weight 가 학습이 가능하도록 설정하였다.

코드는 python pytorch 환경에서 개발되었으며 linux 운영체제에서 11GB 2080 Ti GPU 를 사용해 학습하였다. Total epoch 은 2.5k, Train batch size 는 512, Optimizer 는 Adam, lr 는 0.0001 을 사용하였고 별도의 scheduler 는 사용하지 않았다. FID score 를 측정시 60000 장의 이미지를 sampling 했다. 학습 및 샘플링 코드는 <https://github.com/inooni/COSE474> 에서 확인 가능하다.

4. Future Direction

본 프로젝트는 총 Timestep 이 작은 상황에서 더 나은 샘플링이 가능한 TP Sampling 을 제안하였고, 이를 실험을 통해 성능이 좋아짐을 확인하였다. 하지만 아직

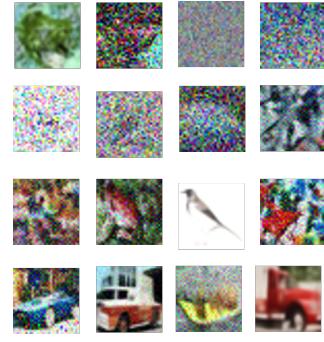


Figure 5. TP Sampling 을 통해 생성낸 이미지 ($T = 10$)

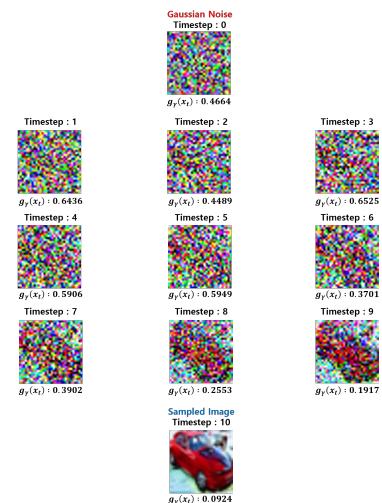


Figure 6. TP Sampling Process

Time Predictor 의 정확도가 너무 떨어져 제안한 method 의 신뢰도가 떨어진다. 또한 Figure 2 에서 볼 수 있듯이 Time-Predictor 학습 도중 발산하는 경우가 많아 안정적인 학습이 불가능하다. 이에 lr scheduler 혹은 새로운 구조를 사용하여 Time-Predictor 의 정확도를 올리는 것이 최우선적으로 이루어져야 한다. 추가로, 현재는 단순히 MSE Loss 의 루트값으로 Time-Predictor 의 오차 범위를 계산했지만, t 에 따라 모델의 정확도가 다를 수 있기 때문에 이에 대한 통계적인 분석이 필요해 보인다.

Time-Predictor 의 성능이 어느정도 향상되었다면, Original Sampling 에서의 x_t 들을 Time-Predictor 으로 분석함으로써 앞서 본 프로젝트에서 문제 삼은 x_t 와 $p_t(x_t)$ 사이 차이가 얼마나 있는지 확인해야 한다. 그 후로는, time interval 이 고정되지 않은 또 다른 Sampling Method 를 고안해보거나, 혹은 더 큰 T 에 대해서도 제안한 method 가 이점을 가져다 주는지 확인해야 할 것이다.

References

Chen, T., Zhang, R., and Hinton, G. Analog bits: Generating discrete data using diffusion models with self-conditioning, 2022.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images, 2009.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

Vincent, P. A connection between score matching and denoising autoencoders. 2011.